

10/10/00  
JCS94 U.S. PTO

10-12-00

A

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

ATTORNEY DOCKET NO. 0544MH-40014

In re Application of:

DANIEL BROWN et al.

Serial No To Be Assigned

Filed: Herewith

For: RULES-BASED NOTIFICATION SYSTEM

JCS920 U.S. PTO  
09/686446  
10/10/00

**TRANSMITTAL**

**BOX: Patent Application**

Hon. Commissioner of Patents  
and Trademarks  
Washington, D.C. 20231

Sir:

Enclosed find:

1. Transmittal with Certificate of Mailing
2. Patent Application
3. Informal drawings
4. Our firm check in the amount of \$710.00
5. Our return postcard, which we would appreciate your date-stamping and returning to us upon receipt.

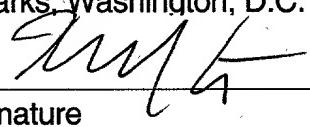
**CERTIFICATE OF EXPRESS MAIL 37 C.F.R. § 1.10**

Express Mail No. **EL609930870US**

I hereby certify that this correspondence is being deposited with the United States Postal Service as "Express Mail Post Office to Addressee" in an envelope addressed to:  
Commissioner of Patents and Trademarks, Washington, D.C. 20231 on the date below.

10 Oct 2000

Date

  
Signature

The Commissioner is hereby authorized to charge any additional fees which may be

required, or credit any overpayment to Deposit Account No. **50-1060**.

Please conduct all correspondence in the above matter with:

Kenneth C. Hill  
HILL & HUNN LLP.  
201 Main Street, Suite 1440  
Fort Worth, Texas 76102-3105  
(817) 332-2113  
(817) 332-2114 (fax)

JC920 U.S. PRO  
09/686446



10/10/00

Respectfully submitted,

A handwritten signature in black ink, appearing to read "Ken Hill".

---

Kenneth C. Hill, Reg. No. 29,650  
HILL & HUNN, LLP.  
201 Main Street, Suite 1440  
Fort Worth, Texas 76102  
817/332-2113

Attorney for Applicant

# **SPECIFICATION**

Docket No. 0544MH-40014

DRAFT - NOT FOR FILING

**TO ALL WHOM IT MAY CONCERN:**

**BE IT KNOWN** that we, Daniel Brown, William Lance Eason, and George A. Thompson, residing in the State of Texas, have invented new and useful improvements in a

## **RULES-BASED NOTIFICATION SYSTEM**

of which the following is a specification:

## CROSS-REFERENCE TO RELATED APPLICATION

1        This application claims the benefit of United States Provisional Application  
2        No. 60/158,732, filed October 11, 1999. This application also contains matter in  
3        common with copending US application Serial No. \_\_\_\_\_, filed on even  
4        date herewith, titled CUSTOMIZABLE USER WINDOW, which is incorporated by  
5        reference.

## BACKGROUND OF THE INVENTION

6        1.        Field of the Invention:

7        The present invention relates generally to communications for electronic  
8        computer systems, and more specifically to a system and method for notifying a  
9        user that selected events have occurred within the system.

10      2.        Description of the Prior Art:

11      As computer systems and data communications become increasingly  
12     widespread, more and more users are able to perform a high percentage of their  
13     daily tasks over various communication networks. For example, many users are  
14     able to obtain key information in a timely manner through the use of interconnected  
15     software and hardware systems.

1       Because of the great wealth of information available, it often becomes time  
2 consuming for a user to keep up with items and events of interest. In some cases,  
3 a dedicated window may be included on a user's desktop to provide information  
4 about high priority items. For example, a streaming stock quote ticker may be  
5 resident on a user's desktop to enable them to keep up with selected stock market  
6 information. However, the number and variety of sources of potentially interesting  
7 information is so large that this type of dedicated approach can work for only a  
8 small number of sources.

9       Often, a user is required to access a database or other source of information  
10 in order to determine whether an event of interest has occurred. This can be  
11 relatively time consuming, and often results in undesirable delays if the user is  
12 unable to check the required sources often enough.

13       Another approach is to automatically send notifications directly from an  
14 application to a user when selected events occur. This requires coding of the  
15 necessary handling and routing modules within the application. This can become a  
16 significant burden from the application writer's standpoint, and can lead to  
17 inconsistencies in the way message notifications are presented to users.

18       It would be desirable to provide a system and method for automatically  
19 notifying users upon the occurrence of selected events. It would further be  
20 desirable for such notification system to be able to handle different types of events  
21 and present notifications to users in different ways.

## SUMMARY OF THE INVENTION

1        In accordance with the present invention, a system and method for providing  
2 notifications to a user includes an alert manager. The alert manager receives  
3 notifications of events incurring within various executing applications, and utilizes  
4 rules to determine when alert notifications should be sent to one or more specified  
5 users. In addition, operation of batch jobs can determine changes in state of  
6 selected data, which generate events for the alert manager in addition to those  
7 generated directly by an executing application. Alert notifications to users can be  
8 sent to the user via any of several communications channels.

DRAFT DRAFT DRAFT DRAFT DRAFT

## BRIEF DESCRIPTION OF THE DRAWINGS

1       The novel features believed characteristic of the invention are set forth in the  
2 appended claims. The invention itself however, as well as a preferred mode of use,  
3 further objects and advantages thereof, will best be understood by reference to the  
4 following detailed description of an illustrative embodiment when read in  
5 conjunction with the accompanying drawings, wherein:

6              Figure 1 is a high-level block diagram of a system in accordance with the  
7 present invention;

8              Figure 2 is a block diagram indicating additional details of the system of  
9 Figure 1;

10          Figures 3A and 3B illustrate operation of different types of event generation  
11 techniques;

12          Figure 4 shows the type of information maintained by an event router in  
13 accordance with the present invention;

14          Figure 5 is a block diagram illustrating an operation of an alert manager;

15          Figure 6A illustrates one preferred embodiment for using a rule filter with  
16 event handlers; and

1       Figure 6B illustrates an alternative embodiment for using a rule filter with  
2       event handlers.

## **DESCRIPTION OF THE PREFERRED EMBODIMENT**

1        As will be understood by those skilled in the art, the present system and  
2 method can be used with nearly any computer system in which events occur during  
3 execution of an application, or in which data is changed. In the preferred  
4 embodiment, the operating applications are of a type which generate messages  
5 indicating the occurrence of selected events. However, as described below, the  
6 system will also work with applications which do not generate such events, and is  
7 adaptable to nearly any type of computer application.

DRAFTSIGHT DOCUMENT

8        Referring to Figure 1, a preferred embodiment of the inventive system and  
9 method is illustrated generally as reference number 10. A plurality of applications  
10 12 operate concurrently, and may be located on a single piece of hardware, or  
11 spread among numerous machines. The applications 12 need not operate in  
12 conjunction with one another, but may be a wide variety of selected applications, of  
13 any type, which perform data processing functions.

14      As will be described in more detail in connection with Figure 2, applications  
15 12 generate events 14, which are forwarded to an event router 16. Events 14 are  
16 basically messages indicating that some type of predetermined event has occurred.  
17 As defined further in connection with Figure 2, these messages may be generated  
18 directly by applications to indicate that an event has occurred in that application.  
19 For example, an application which is used to change the price of a product in a

1 database can generate an event message indicating the product number and new  
2 price for the product. In addition, as described below, certain types of applications  
3 12 may periodically scan persistent information, such as that stored in databases,  
4 to determine whether data has changed. These applications can also generate  
5 events showing that changes of the selected type have occurred.

6 Event router 16 is described in more detail in connection with Figure 4, but is  
7 generally an application which receives incoming events 14, and routes them to  
8 recipients which have registered to receive events of this type. For example, if a  
9 user or another application has registered to receive notification of price change  
10 events, the event router 16 will maintain this information. When an incoming price  
11 change event is received by event router 16, it will then be forwarded to all  
12 applications which have been registered to receive this event.

13 Event router 16 therefore acts as a central routing point for messages. It is  
14 not necessary for applications 12 to know, or be able to find out, who will receive  
15 notifications of events. Instead, applications 12 simply generate defined events and  
16 send all of them to event router 16. Event router 16 handles the task of sending the  
17 events to as many recipients as are registered to receive them.

18 Some events 18 are routed to various types of event handlers 20, which  
19 then use the incoming event message to process data. In many cases, event  
20 handlers 20 will also be classified as applications 12. In this application, event

1    handlers 20 are generally any applications which make some use of the information  
2    provided by incoming event messages 18.

3           Some events 22 are routed to an alert manager 24, which is a particular type  
4    of event handler. As described further in connection with the remaining figures,  
5    alert manager 24 handles incoming events by determining whether any incoming  
6    event requires that a notification, or alert 26, to be forwarded to a user 28. Alert  
7    manager 24 utilizes a set of rules to determine to whom, and when, notifications  
8    should be made to a user 28.

DRAFT DRAFT DRAFT DRAFT

9           Referring to Figure 2, the types of applications 12 which generate events  
10   may be classified generally into two categories. The first category is referred to  
11   herein as business objects 30, which may include within them state machines 32  
12   and similar operating modules. Business objects 30 generate “explicit” events,  
13   meaning that the code of the business object application explicitly generates an  
14   event to be sent to event router 16 as events occur. This capability must be  
15   programmed into business objects 30 and their state machines 32.

16           Explicit events are useful, in the present invention, for indicating when a  
17   single event has occurred. For example, when a sale is made, or a product  
18   changes price, or a new product becomes available, if the corresponding business  
19   objects are properly programmed events will be generated. These can be picked  
20   up by alert manager 24, and used to generate notifications.

1           A second kind of application that can generate events is referred to herein  
2 generally as "batch jobs." These jobs are applications that, generally, periodically  
3 check persistent data, such as data stored in a database, and look for changes that  
4 may have occurred. For example, if an application which enters new products into  
5 a database is not one which has previously been coded as a business object, to  
6 generate explicit events on this occurrence, a batch job 34 can periodically scan a  
7 product database and determine when new products have been added. Events  
8 which are discovered by such a comparison between a previous state of an object,  
9 in a persistent memory, with the current state are referred to herein as "implicit  
10 events."

11           Use of batch jobs to scan data looking for implicit events is useful both for  
12 events which occur over time, and for use with applications which are not already  
13 coded to generate the desired explicit events.

14           Event router 16 can route events to two different types of handlers. These  
15 can be categorized generally as synchronous handlers 36 and asynchronous  
16 handlers 38. Events which are intended for use by asynchronous handlers are  
17 preferably placed into a queue 40, from which they are later withdrawn by various  
18 asynchronous handlers 38. More than one queue 40 can be provided, and  
19 registration of a handler with the event router, to be notified of a particular type of  
20 event, will indicate whether the handler is synchronous, or asynchronous, and in  
21 the latter case indicate which queue is used for the event message.

1       Figures 3A and 3B illustrate the operational difference between explicit and  
2 implicit events. In Figure 3A, dataflow occurring upon an explicit event is shown. In  
3 this example, a user running an application enters a new product, number 789, and  
4 identified as a "BOAT", into the system.

5           When the user desires to add the product, procedure add-product is  
6 executed 42. This procedure passes the new product number, '789', and the new  
7 item description, 'BOAT', to business object 44. When the new product is set up by  
8 business object 44, an event 46 is generated and sent to event router 16. Event 46  
9 indicates that a new product has been added, and passes the relevant parameters  
10 as part of the message.

11          Referring to Figure 3B, addition of the same product is illustrated when no  
12 business object is available to generate an explicit event. Instead, it will be  
13 assumed that an appropriate batch object 34 runs periodically to check a persistent  
14 data base. In this example, the batch job runs at 9:00 o'clock, and detects a table  
15 in the data base which shows two products, numbers 123 and 456. This  
16 information is retained by the batch object for future use. At a subsequent time,  
17 9:08 in this example, a user executes the add-product procedure as was the case  
18 in Figure 3A. However, business object 50 is one which does not generate an  
19 explicit event message when the data base is updated. Thus, no message is sent  
20 to event router 16 at 9:08.

1       At 10:00, in this example, the batch job executes again. At this time, the  
2 information it collects is shown in table 52 and includes three products. By  
3 comparing the present table 52 with earlier table 48, the batch job determines that a  
4 new product has been added to the data base. The batch job then generates an  
5 implicit event showing that the new product has been added, and sends it to router  
6 16.

7       Depending upon the implementation, it may be desirable for implicit events  
8 to carry a flag distinguishing them from explicit events to assist either the router or  
9 handler in determining how to treat the message. However, in other cases, it may  
10 not be necessary to distinguish between implicit and explicit events; only the fact  
11 that an event occurs would be of interest to the handlers.

12       Figure 4 indicates the type of information retained by event router 16 in order  
13 to route events which it receives. The information is shown in Figure 4 as a table,  
14 but may be stored internally within event router 16 in any desirable form.

15       Referring to Figure 4, the table indicates that event router 16 stores a list of  
16 types of events which are to be routed. This table is preferably dynamic, and can  
17 be added to as various handlers register with event router 16. In the example of  
18 Figure 4, EVENT TYPE 1 has three recipients, R1, R2 and R3, registered to  
19 receive a copy of this type of event. Thus, when an event of EVENT TYPE 1 is  
20 received by event router 16, recipients R1, R2 and R3 in turn receive a copy of the  
21 event message. Identification of the recipient indicates where the message will be

1 sent, and any other particular conditions under which it should be forwarded. If any  
2 recipients R1, R2 or R3 are synchronous handlers, the event message will be  
3 copied and forwarded to them immediately. If any of these recipients are  
4 synchronous events, the registration of such event will indicate a queue into which  
5 a copy of the event message should be placed.

6 Figure 4 shows five different event types, with various recipients registered  
7 to receive them. Some event types, such as EVENT TYPE 3 in this case, may not  
8 have any registered recipients. In such case, any events of this type are not routed  
9 to a handler. In most cases, however, one or more recipients will be registered to  
10 receive an event, and event router 16 will forward copies of the event message to  
11 all registered recipients.

12 By use of the event router, recipients can register to receive messages of  
13 various types without having any impact on the business objects which generate  
14 these events. Instead, the events are consistently passed to event router 16, from  
15 which any handler which needs to be copied on the event message receives its  
16 forwarded copy. Each handler will then deal with a message according to its  
17 internal logic.

18 Figure 5 is a block diagram illustrating functioning of the alert manager 24.  
19 As described above, alert manager 24 is preferably an asynchronous handler and  
20 therefore receives its events from a queue.

1        Within alert manager 24 are two major portions. These include a rules  
2 portion 60, and a notifications portion 62. Rules portions 60 contains a large  
3 number of rules defining when events are to be acted upon. Each user who  
4 desires to receive alert notifications from alert manager 24 will register with the alert  
5 manager, and define the conditions under which that user wishes to receive a  
6 notification. Rules are generally conditional statements which define where the  
7 notification is to be generated.

8        In the example show in Figure 5, two rules are provided. The conditional for  
9 the first rule provides that a product equals ("BOAT"), and a price less than  
10 \$7,500.00. If an event occurs which makes this conditional TRUE, a corresponding  
11 notification 62 will be generated to the user requesting this alert.

12        In this example, a price-change event 64 is passed to alert manager 24 by  
13 its associated queue, the rules portion 60 compares the context data passed along  
14 with the message with its rule conditional portions. The comparisons are performed  
15 by a rules engine 66, which receives rules and context data from rules portion 62  
16 and returns Boolean values of true or false depending upon the evaluation of the  
17 conditional. In this example, rules engine 66 returns a value of TRUE for the first  
18 rule, because the product is "BOAT" and the price is less than \$7,500.00.

19        When the condition of a rule is satisfied, the corresponding notification is  
20 then executed and sent to the user 28. Notifications can involve sending a  
21 message by a computer communications network to a user's desktop. Notifications

1 can also include sending of a message via phone, pager, fax, or any other desired  
2 technique. The contents of the message are defined by the notification portion of  
3 the rule. Preferably, a fairly large number of preselected message types will be  
4 available, so that a user will be able to select from these messages. The messages  
5 can include textual information, and can include as parameters some or all of the  
6 context information included with event 64. In the example of Figure 5, an e-mail  
7 message can be sent to the user who registered Rule 1 indicating that a price  
8 change for BOAT has occurred, and indicating the new price for the BOAT. A call  
9 can be made to a pager at the same time; any desired number of notifications can  
10 be made.

11 This rule base system allows users to register with the alert manager 24,  
12 defining logical conditions under which alerts of various types will be sent. This  
13 frees the user from having to check for events or changed conditions individually;  
14 this is done automatically by the rules set up in the alert manager. Users can  
15 determine how these messages are to be sent. E-mail would be one typical type of  
16 message; users may also provide for one or more notification windows to be  
17 generated upon their desktop for the sole purpose of receiving alert notifications.

18 By setting up and registering different types of alerts with a central system, a  
19 user can be notified regarding a wide variety of events which would otherwise take  
20 too much time and effort to profitably be viewed. Upon receiving one of these  
21 alerts, the user can, if she so desires, take a corresponding action.

1       The method described above can be generalized and modified to provide  
2 different functionality. Referring to Figure 6A, a diagram is shown of a system  
3 which uses a rule filter similar to the rules portion of an alert manager 24 for event  
4 handlers in general. When business object 30 generates an event 14, it is  
5 communicated to event router 16. Event router 16 then sends copies of event 70,  
6 72 to event handlers 74 and 76 which have been previously registered with event  
7 router 16. Event handlers 74, 76 each have a rule filter 78, 80, respectively, which  
8 uses conditionals in essentially the same manner as described in connection with  
9 Figure 5. These rule filters allow for initial filtering of events by event handler 74, 76  
10 as part of the determination of which events to respond to.

11       An alternative approach is shown in Figure 6B, in which a combined  
12 router/filter includes event router 16 and rule filter 84. When business object 30  
13 sends an event 14 to event router 16, rule filter 84 is checked to see whether any  
14 rules apply to such event. If a conditional of a rule is met, the functional portion of  
15 the rule, directing events to particular handlers, is then executed. For example,  
16 when event 14 is sent to event router 16, the corresponding copy is not  
17 automatically sent to both event handlers 86, 88. Instead, the context data included  
18 with event 14 is evaluated as described in connection with Figure 5 within rule filter  
19 84, and events are only sent to handlers for which the rule conditional evaluates  
20 true. In this example, event copy 90 is forwarded only to event handler 88 because  
21 no corresponding conditional was satisfied which would cause event 14 to be  
22 routed to event handler 86.

1        Inclusion of rule filter 84 with event handler 16 allows for several changes in  
2    the system. Initially, router/filter 82 must be more complicated, and allow handlers  
3    to register rules with filter 84 as well as simply being added to a list in event router  
4    16. Preferably, rules in filter 84 are checked only for handlers which are registered  
5    for a particular event type, so that only a few rules need to be checked when an  
6    incoming event is received.

7        Inclusion of rule filter 84 with event router 16 increases the complexity of the  
8    code associated with the event router, but decreases communications to event  
9    handlers. Only events which satisfy a predefined criteria are passed to an event  
10   handler by rule filter 84. In addition to lessening network traffic, this approach  
11   simplifies all of the event handlers. Each event handler may be simplified by  
12   deleting any initial rule filter used to handle incoming events; instead, this filtering is  
13   performed by rules registered with rule filter 84.

14       The system and method described above provide a simplified method for  
15   generating user alerts in response to various events taking place within  
16   applications. By sending all events to a central location, an alert manager can  
17   register for any desired type of event and, in response to logical conditionals being  
18   satisfied, send alert notifications to any registered users. Because the system can  
19   handle both explicit and implicit events, it is suitable for use with existing  
20   applications which do not generate events suitable for routing through event router  
21   16, or which do not generate events of interest to a particular user.

1       Users gain the ability to select types of events for which they wish to be  
2   notified. When these events occur, notification occurs automatically. The user  
3   need not periodically review various locations to determine whether anything has  
4   happened. Users can select from among types of events, and the applications  
5   writers can determine which events to actually make available. For example, a  
6   user can register with the alert manager 24 to be notified whenever a new product  
7   is introduced by a competitor; automatic notifications will be provided whenever this  
8   occurs, assuming the events are available as either explicit or implicit events.

9       Preferably, users will be able to select from one or more menus of choices  
10   provided by the system. This simplifies the task of registering to receive alerts, and  
11   ensures a certain uniformity of notifications across all users. The conditionals used  
12   in the alert rules can be provided as templates, with the user selecting the form of  
13   the conditional and any particular values to be used. In this manner, the user can  
14   easily provide a set of alert rules to meet his or her needs.

15       As described previously, the alert notifications themselves can be provided  
16   in any available format supported by the system. Notification may be by e-mail or  
17   other electronic messaging as known in the art. By sending appropriate messages  
18   to any type of intermediate interface devices, messages such as pages or  
19   telephone alerts can also be made. Because the alert notification message and its  
20   type are maintained in tables in the alert manager, addition of a new technology is  
21   easily made to the alert system. All that is necessary is to provide that a selected  
22   message be sent to an appropriate handler from the alert manager, and the  
23   message can be sent to the registered user.

1        While the invention has been particularly shown and described with  
2 reference to a preferred embodiment, it will be understood by those skilled in the art  
3 that various changes in form and detail may be made therein without departing from  
4 the spirit and scope of the invention.

What is claimed is:

- 1 1. A system for generating notification messages, comprising:
  - 2 a plurality of applications, each application generating event messages;
  - 3 an event router in communication with the applications, wherein the router
  - 4 receives the event messages;
- 5 an alert manager in communication with the router for receiving a selected
- 6 subset of the event messages; and;
- 7 a plurality of rules within the alert manager, each rule having an
- 8 associated notification, and each rule registered to act upon selected events
- 9 within the selected subset of event messages;
- 10 wherein, upon receiving a message at the alert manager, all rules
- 11 registered to act upon the message are evaluated, and for all rules that evaluate
- 12 to True, sending the associated notifications.

1    2.    The system of Claim 1, wherein at least one of the plurality of applications  
2    comprises:

3            a database having data which can be changed; and

4            a job that periodically examines the database, and generates an event  
5            message when selected data items in the database have changed from a  
6            previous examination.

1    3.    A system for generating notifications, comprising:

2                a plurality of applications that generate event messages;

3                a router in communication with the applications;

4                a plurality of event handlers in communication with the router; and

5                a set of rules attached to the router, each rule associated with at least one

6                event message;

7                wherein, when an event message is received, the associated rules are

8                evaluated, and for rules that evaluate to True, the event message is forwarded to

9                a related event handler.

1    4.    The system of Claim 3, wherein at least one of the plurality of applications

2                comprises:

3                a database having data which can be changed; and

4                a job that periodically examines the database, and generates an event

5                message when selected data items in the database have changed from a

6                previous examination.

- 1 5. A method for generating notifications, comprising the steps of:
- 2 generating a event message within an application;
- 3 sending the event message to a router;
- 4 within the router, registering recipient handlers to be notified upon receipt
- 5 of selected event messages;
- 6 within the handlers, providing a set of rules to be evaluated, with each rule
- 7 having a corresponding action to be executed if the rule evaluates True; and
- 8 when an event message is received by a handler, evaluating a subset of
- 9 rules, and executing any actions for rules that evaluate True.
- 1 6. The method of Claim 5, wherein an action to a rule comprises sending a
- 2 message to a user.
- 1 7. The method of Claim 5, wherein the application comprises:
- 2 a database having data which can be changed; and
- 3 a job that periodically examines the database, and generates an event
- 4 message when selected data items in the database have changed from a
- 5 previous examination.

## **ABSTRACT**

1       A system and method for providing notifications to a user includes an alert  
2 manager. The alert manager receives notifications of events occurring within  
3 various executing applications, and utilizes rules to determine when alert  
4 notifications should be sent to one or more specified users. In addition, operation  
5 of batch jobs can determine changes in state of selected data, which generate  
6 events for the alert manager in addition to those generated directly by an  
7 executing application. Alert notifications to users can be sent to the user via any  
8 of several communications channels.

FIG 1

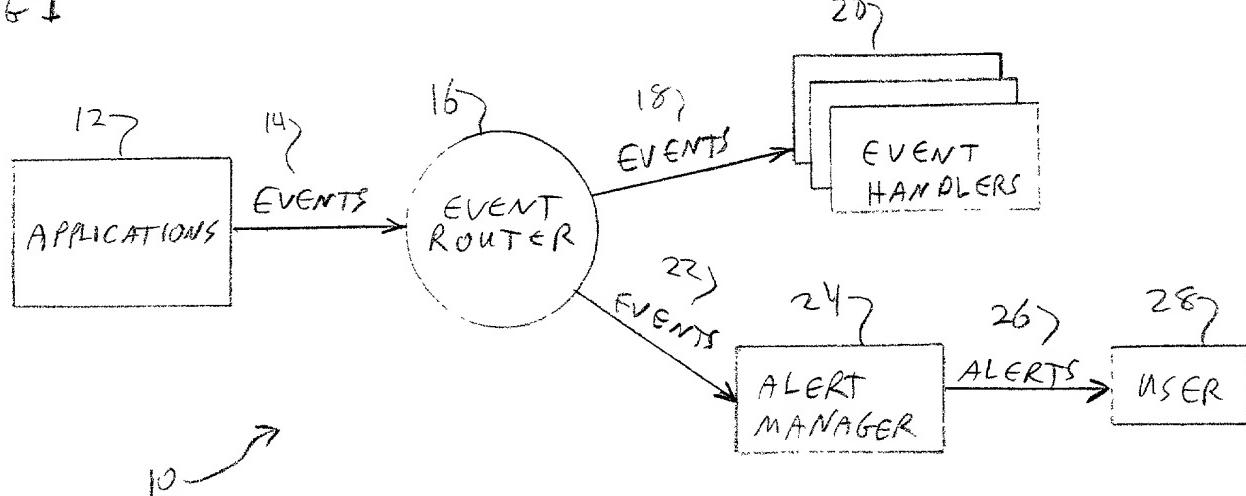
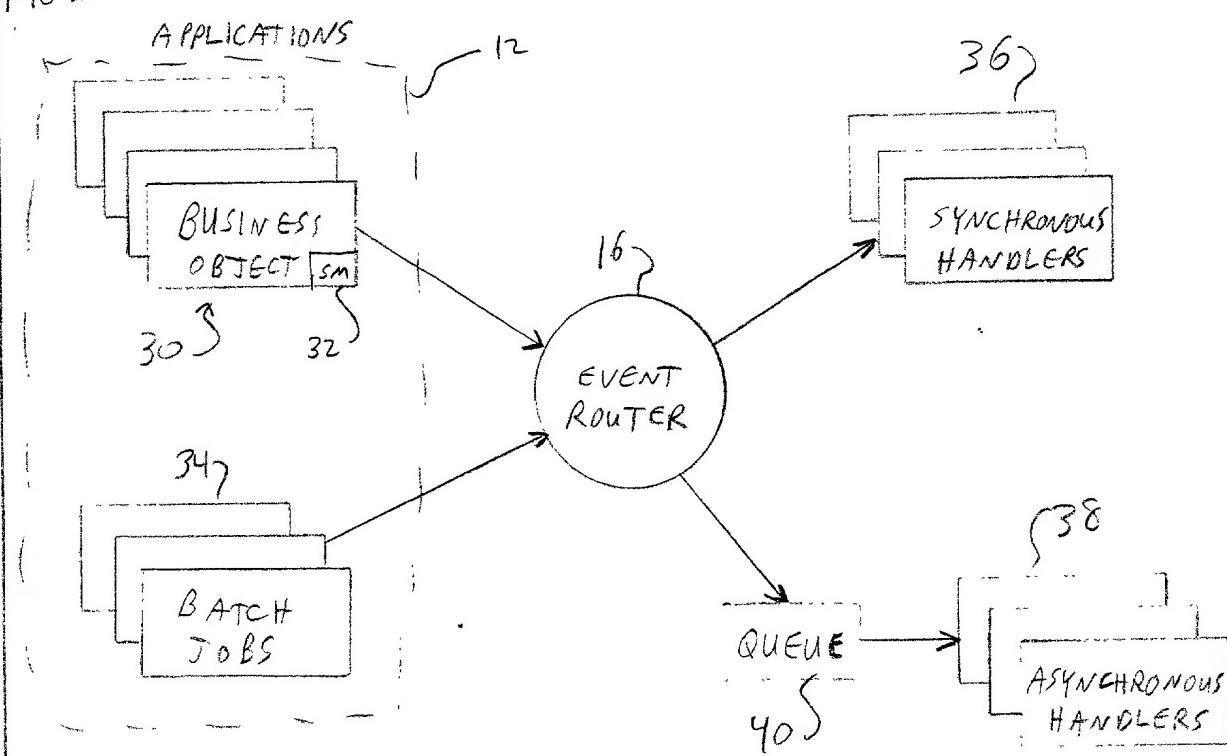


FIG 2



0544MH

40014

2

FIG 3A

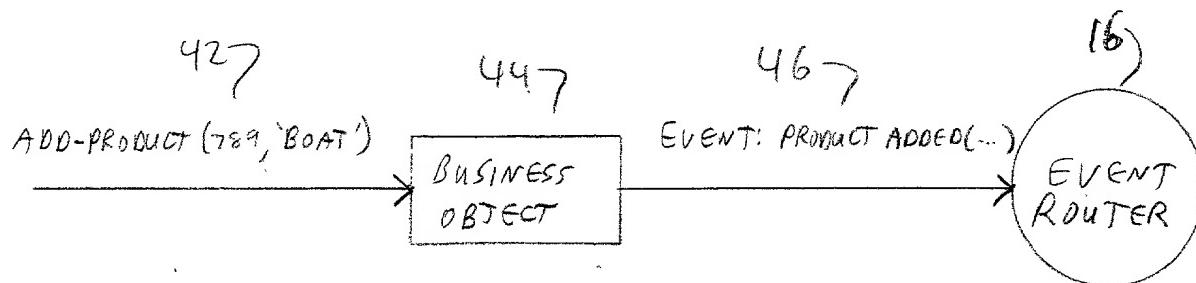


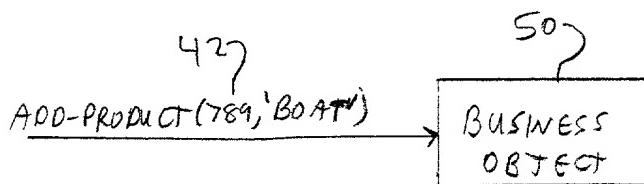
FIG 3B

9:00

PRODUCTS	
123	CAR
456	TRUCK

48

9:08



10:00

PRODUCTS	
123	CAR
456	TRUCK
789	BOAT

52

EVENT: PRODUCT ADDED(...)

54



FIG 4

EVENTS	RECIPIENTS
EVENT TYPE 1	R1, R2, R3
EVENT TYPE 2	R2, R4
EVENT TYPE 3	∅
EVENT TYPE 4	R5
EVENT TYPES	R2, R5
⋮	

22-141 50 SHEETS  
22-142 100 SHEETS  
22-144 200 SHEETS



DO NOT COPY - 2003 © AMPAD CORP.

FIG 5

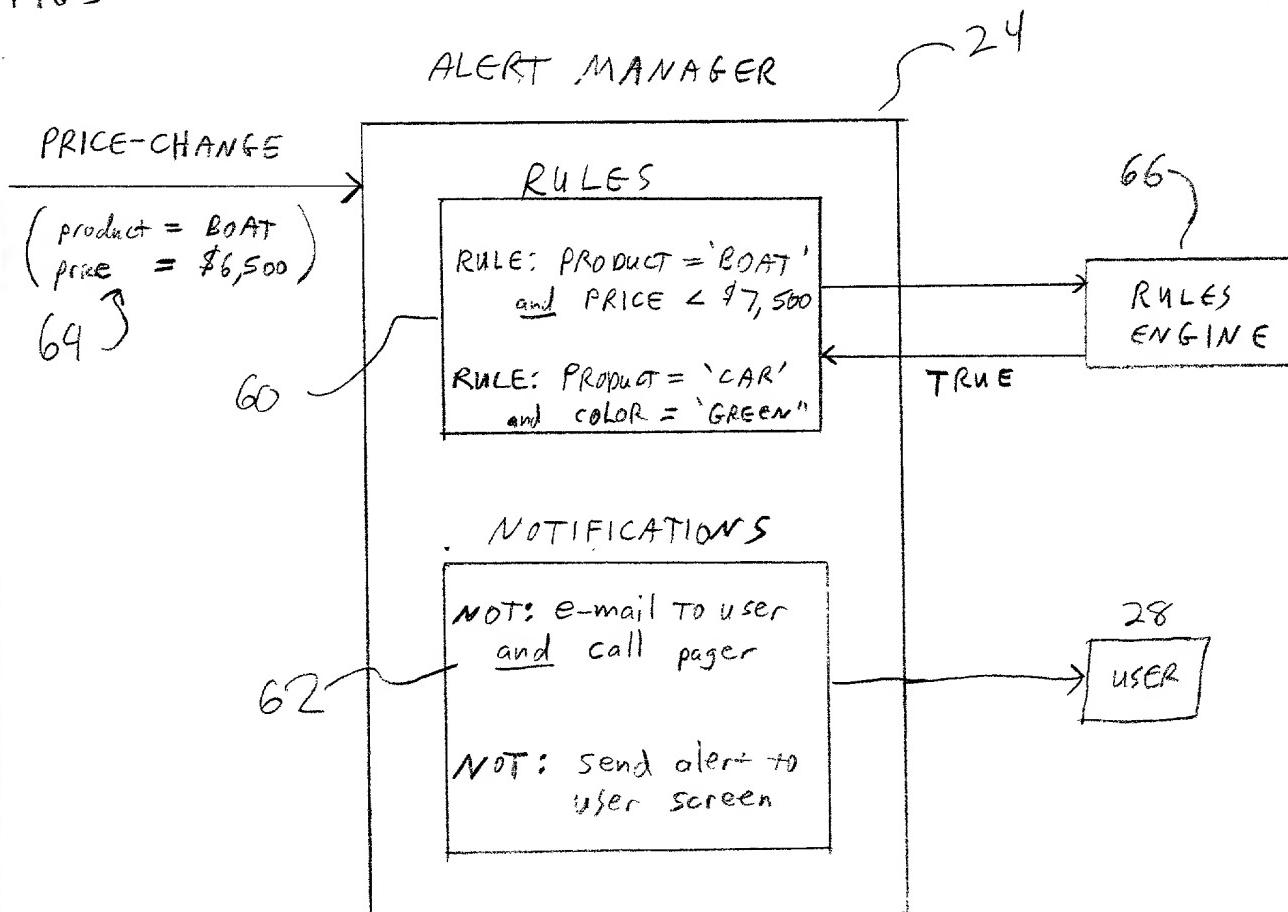
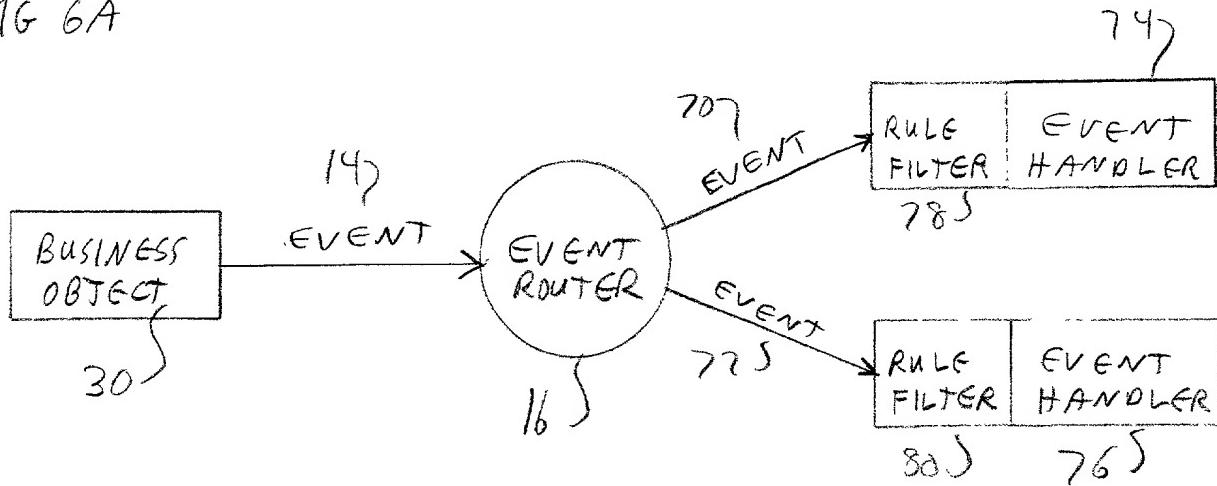


FIG 6A



22-141 50 SHEETS  
22-142 100 SHEETS  
22-144 200 SHEETS



00000000000000000000000000000000

FIG 6B

